Massiv-parallele numerische Simulation auf Grafikkarten: The poor man's supercomputer?!

Krafczyk, Manfred

Veröffentlicht in:
Jahrbuch 2010 der Braunschweigischen Wissenschaftlichen Gesellschaft, S.139-142

J. Cramer Verlag, Braunschweig

Massiv-parallele numerische Simulation auf Grafikkarten: The poor man's supercomputer?!

MANFRED KRAFCZYK

Museumstraße 6, D-38100 Braunschweig

Die Effizienz und Genauigkeit computergestützter Analysen und Verhaltensprognosen komplexer Systeme in Natur und Technik basieren auf Fortschritten der (numerischen) Mathematik, der Informatik und der technologischen Fachdisziplinen. In dieser Zusammenfassung wird über neue Entwicklungen im Bereich spezieller Hardware (General Purpose Graphics Processing Units, GPGPUs) und ihre Verwendung zur Berechnung komplexer Ingenieurprobleme berichtet. Obwohl über die letzten vier Dekaden die Leistungsfähigkeit der Mikroprozessoren gemessen an der Anzahl von Fließkommaoperationen pro Sekunde (FLOPS) gemäß dem Gesetz von Moore exponentiell zugenommen hat, reicht die Leistungsfähigkeit eines einzelnen Prozessors bei weitem nicht aus, um beispielsweise dreidimensionale und zeitabhängige Strömungsprobleme mit hinreichender Genauigkeit in akzeptabler Zeit zu simulieren. Um diese Problematik zu entschärfen, wurden verschiedenste numerische Verfahren entwickelt, die nach dem *divide et impera*-Prinzip eine mehr oder weniger gekoppelte, parallele Bearbeitung des Gesamtproblems durch Zerlegung in Teilprobleme ermöglichen und so durch die Verwendung von sog. Parallelrechnern bestehend aus bis zu Hunderttausenden von vernetzten Prozessoren (Central Processing Units, CPUs) und einer Gesamtleistungsfähigkeit von ca. einem PetaFLOP (10^{15} FLOPS) eine substantielle Reduktion der Rechenzeit ermöglichen. Solche Großrechner sind jedoch weder für typische kommerziell eingesetzte Berechnungsmethoden geeignet noch für die industrielle Praxis aus Kostengründen relevant. In den letzten Jahren wurde neben den Entwicklungen im Bereich von CPUs insbesondere Anstrengungen unternommen, die Leistungsfähigkeit von GPUs zu erhöhen. Eine spezielle Entwicklungslinie stellen hier die GPGPUs dar, die nicht mehr primär zur Visualisierung dienen, sondern die Funktion numerischer Co-Prozessoren einnehmen. Solche GPGPUs bestehen ihrerseits aus hunderten von so genannten Kernen, auf denen ein Gesamtproblem parallel bearbeitet werden kann.

---

* Der Vortrag wurde am 08.10.2010 in der Klasse für Ingenieurwissenschaften der Braunschweigischen Wissenschaftlichen Gesellschaft gehalten.

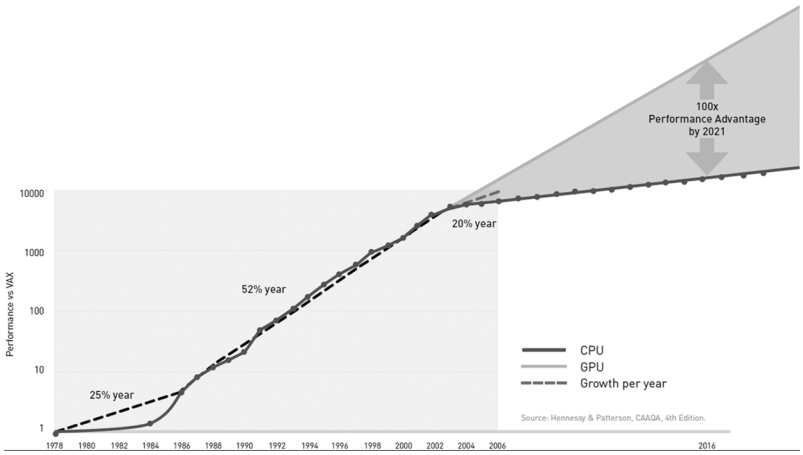

Abb. 1: Extrapolierte Prognose der Leistungsfähigkeit zukünftiger CPUs und GPUs [1].

Falls die problemspezifischen numerischen Algorithmen passend für das den GPGPUs zugrundeliegende Single Instruction Multiple Data (SIMD) Programmierparadigma implementiert werden können, ist man somit in der Lage, aus einem leistungsfähigen PC unter Verwendung von bis zu vier parallel operierenden GPGPUs eine Leistung von mehr als einem TeraFLOP (10^{12} FLOPS) extrahieren zu können [2]. Darüber hinaus können GPGPUs auch über ein entsprechendes Netzwerk gebündelt zum Einsatz kommen wie z.B. im Cluster Ludwig an der TU Braunschweig, wo 96 GPGPUs verteilt an einem Problem eine Gesamtleistung von bis zu 40 TeraFLOP erbringen, wobei in Bezug auf Anschaffungspreis und Unterhaltskosten eine Effizienzsteigerung von bis zu einer Größenordnung in Bezug auf konventionelle CPU-basierte Systeme möglich ist. Insbesondere dieses Preis-Leistungsverhältnis macht GPU-basierte Systeme für einen breiten Kreis von Industrieanwendern interessant, da somit erstmalig die Durchführung von sehr aufwändigen Simulationen möglich ist, ohne massiv in entsprechende Hardware und Systemadministration für einen Parallelrechner investieren zu müssen. Voraussetzung zur Nutzung GPU-basierter Simulationen ist allerdings eine Anpassung der problemspezifischen Berechnungsverfahren an die GPU-spezifische Speicher- und Prozessstruktur, welche durch spezifische Entwicklungsumgebungen und Standards [3, 4] deutlich erleichtert werden.

Ein Modellansatz, der auf GPGPUs sehr effizient zu implementieren ist, basiert auf dem sog. Gitter-Boltzmann-Ansatz [5], bei dem durch sukzessive Vereinfachungen der Boltzmann-Gleichung ein expliziter numerischer Ansatz zur Lösung der schwach kompressiblen Navier-Stokes-Gleichungen abgeleitet werden kann (siehe Abb. 2).

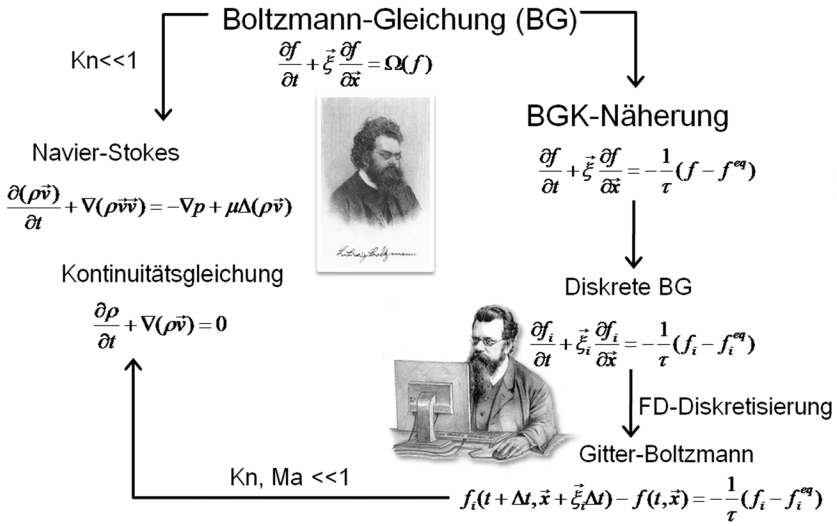

Abb. 2: Ableitung der Navier-Stokes-Gleichung aus der Boltzmann-Gleichung und dem Gitter-Boltzmann-Ansatz.

Exemplarisch sei hier als methodische Validierung der sog. Wellengenerator von Russel [6] aufgeführt, bei dem durch Versenken eines Blockes näherungsweise ein Soliton initiiert wird, dessen Simulation mit GPU-basierten Lattice-Boltzmann-Verfahren in der Dissertation von C. Janßen [7] am iRMB der TU Braunschweig durchgeführt wurde (Abb. 3).

Die zeitabhängige Simulation des gekoppelten Problems einer turbulenten Strömung mit freien Oberflächen und bewegten Rändern mit mehr als 2×10^6 Freiheitsgraden ist auf einer Nvidia C1060 GPU innerhalb weniger Minuten möglich.

Obwohl die Beschleunigung numerischer Berechnungen durch GPGPUs für die hier verwendeten Verfahren mindestens bei einer Größenordnung liegt, ist die Entwicklung und Validierung von Algorithmen optimaler Komplexität (z.B. Mehrgitterverfahren) durchaus noch Gegenstand aktueller Forschung [8], allerdings muss davon ausgegangen werden, dass sich zukünftige Hardwaregenerationen der ExaFLOP-Klasse nicht mehr ohne eine Kombination von CPUs und GPGPUs realisieren lassen [9]. Neben den für dieses ambitionierte Vorhaben umfangreichen Forschungsarbeiten auf dem Gebiet numerischer Methoden und der verteilten Programmierung lässt sich aber jetzt schon eine durch die Verwendung von GPGPUs deutlich gesteigerte Nutzbarkeit moderner dreidimensionaler und zeitabhängiger numerischer Verfahren auch für kleine und mittlere technisch orientierte Unternehmen absehen, denen ein Zugang zu diesen Methoden durch die vergleichsweise unmäßigen Kosten für die Anschaffung und den Betrieb von klassischer paralleler Hardware verwehrt war.

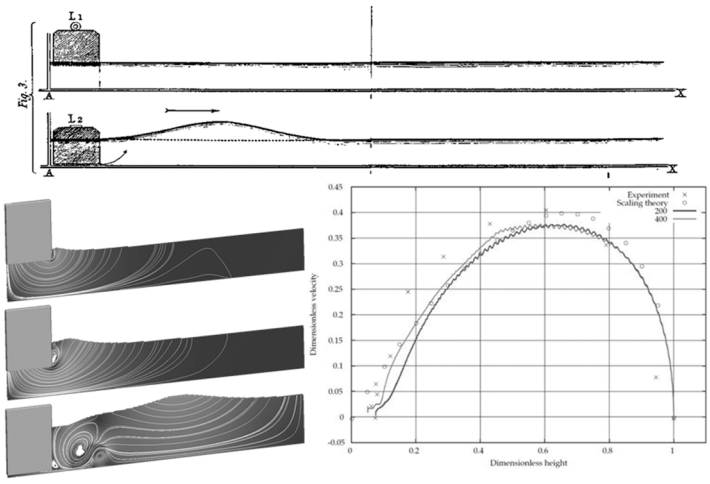

Abb. 3: Zeitlicher Verlauf der Ausbildung einer Welle durch Absenken eines Gewichtes in einem numerischen Wellenkanal [7], rechte Seite: Simulation und Experiment.

Literatur

[1] http://www.nvidia.de/object/LO-tesla-brochure-12-lr.html

[2] Tölke, J. & M. Krafczyk (2008): 'TeraFLOP computing on a desktop PC with GPUs for 3D CFD', International Journal of Computational Fluid Dynamics, **22**(7): 443– 456

[3] http://www.nvidia.de/object/cuda_home_new_de.html

[4] http://www.khronos.org/opencl/

[5] Krafczyk, M., J. Tölke, B. Ahrenholz, S. Bindick, S. Freudiger, S. Geller, C. Janßen & B. Nachtwey (2009): Kinetic Modeling and Simulation of Environmental and Civil Engineering Flow Problems, in Hirschel, E. & E. Krause (Eds.), 100 Volumes of 'Notes on Numerical Fluid Mechanics', Springer, ISBN: 978-3-540-70804-9, 341–350.

[6] Monaghan J.J. & A. Kos (2000): Scott Russell's wave generator. Physics of Fluids **12**: 622–630.

[7] Janß en, C. (2011): Enhanced free surface flow simulations using kinetic methods, Dissertation, Fakultät Architektur, Bauingenieurwesen und Umweltwissenschaften, TU Braunschweig.

[8] Göddeke, D. (2010): Fast and Accurate Finite Element Multigrid Solvers for PDE Simulations on GPU Clusters, Dissertation, Fakultät für Mathematik, TU Dortmund.

[9] Dally, B. (2010): GPU Computing to Exascale and Beyond, Vortrag Supercomputing, http://www.nvidia.com/content/PDF/sc_2010/theater/Dally_SC10.pdf